Detecting Sponsored Recommendations

Subhashini Krishnasamy[1], Rajat Sen[1], Sewoong Oh[2], and Sanjay Shakkottai[1]

[1]The University of Texas at Austin
[2]University of Illinois at Urbana-Champaign

Abstract

With a vast number of items, web-pages, and news to choose from, online services and the customers both benefit tremendously from personalized recommender systems. Such systems however provide great opportunities for targeted advertisements, by displaying ads alongside genuine recommendations. We consider a *biased* recommendation system where such ads are displayed without any tags (disguised as genuine recommendations), rendering them indistinguishable to a single user. We ask whether it is possible for a small subset of collaborating users to detect such a bias. We propose an algorithm that can detect such a bias through statistical analysis on the collaborating users' feedback. The algorithm requires only binary information indicating whether a user was satisfied with each of the recommended item or not. This makes the algorithm widely appealing to real world issues such as identification of search engine bias and pharmaceutical lobbying. We prove that the proposed algorithm detects the bias with high probability for a broad class of recommendation systems when sufficient number of users provide feedback on sufficient number of recommendations. We provide extensive simulations with real data sets and practical recommender systems, which confirm the trade offs in the theoretical guarantees.

1 Introduction

The growth of online services has provided a vast variety of choices to users. This choice exists today in multiple domains including e-commerce with a variety of products, and online entertainment (NetFlix, Pandora). With users having to choose from an overwhelming set of items, recommender systems have become indispensable in easing the information overload and search complexity. Recommender systems are not restricted to retail businesses. A search engine like Google can be viewed as a recommendation engine that helps users find relevant information by ranking the search results according to their search criteria, history and other personal information. Social networking sites like Twitter and Facebook display Tweets and News Feed based on users' past behavior and their connections to other users. News portals like Yahoo! News also present personalized content to online news readers.

Personalized recommender systems serve as an attractive platform for advertisers to reach their targeted consumers. It is now customary to see ads alongside other genuine recommendations in many of the websites that provide recommendation services. One can distinguish these ads from genuine recommendations, for example, by the location of their placement or by their special tags. But recommendation engines are not legally obliged to facilitate such distinction and could possibly serve these ads mixed with genuine recommendations in a manner that renders them indistinguishable to users. Such a *biased recommender system* can have far reaching consequences, including user dissatisfaction with the recommendations [1]. A recent survey by Facebook shows that users find sponsored ads mixed with genuine posts in their News Feed more annoying than the explicit, well-separated ads [2]. Social and political consequences of bias in the context of media and online content have also been studied [3, 4, 5].

Modern recommender systems, in general, consist of two components: *(i)* learn individual preferences from user feedback, and *(ii)* recommend items to users based on the estimated preferences. This combination of learning and recommending is bound to be noisy (the learning phase will *explore* individual preferences typically by presenting "random" recommendations), and several recommendations to users will likely be ineffective. Critically, both noise and bias manifest as bad recommendations to users. However, noise is benign and is a consequence of learning, while bias is systematic and is to be deprecated. Thus, a basic question of interest to the users of such systems is whether or not such a biased recommender system can be detected. This is a broad question, and detecting bias in its most general sense is out of the scope of this paper. We focus on a detecting a specific type of bias where recommendation engines *systematically favor a few items over other better or at least equally good items, contrary to what an objective or unbiased system would do.*

It should be noted that, with most service providers being non-transparent about their recommendation strategies, one cannot hope to know the exact statistical profile of the recommendation engine *a priori*. Therefore, the key is to identify the primary features that can be used to differentiate between the two types without any *a priori* knowledge about the particulars of the recommendation strategy. One could, for instance, consider the average rating or the average number of ineffective recommendations as the performance measure and make a decision based on a threshold parameter. However, as we also demonstrate through simulations, such a basic algorithm based solely on average performance cannot distinguish between deliberate systematic bias and innocuous random errors. This brings us to the key question: *Can we develop a better method to expose a biased recommender system?*

1.1 Contributions of this paper

We say a recommendation engine is *biased*, if it systematically favors a small set of items over other items in the database irrespective of users' preferences. On the other hand, we say that a recommendation engine is *objective*, if it satisfies a simple monotonic property in its recommendations to users – better suited items are given higher priority (in a statistical sense). The primary goal of this paper is then to develop algorithms to answer the following question: *Can a meaningful distinction be drawn between objective and biased recommendation engines?*

BiAD Algorithm: We propose an anomaly detection algorithm that we call *Bi*nary feed-

back *A*nomaly *D*etector (*BiAD*), which uses a statistical approach to identify a *biased* recommendation engine. Under appropriate conditions on the size of the ad-pool, the aggressiveness of the biased recommender system, and the number of users/samples, we show that BiAD correctly (with high probability) distinguishes between objective and biased recommendation engines.

The algorithm leverages user collaboration, and is based on the observation that a *biased* system is typically characterized by the occurrence of a *large number of ineffective recommendations in a small set of items*. On the contrary, giving higher priority to more effective items, as in an *objective* recommender system, precludes such concentration in a small set. Notably, since the users are not aware of the set of items, the BiAD algorithm is adaptive – as the recommender system learns users, the users "learn" the recommender system. Further, our algorithm relies only on binary feedback on the effectiveness of the recommendation. Finally, the BiAD algorithm also works for a large class of recommender systems since our model does not place any constraints on the recommendation engine other than mild statistical conditions. We finally present extensive simulation results that cover various types of recommender systems and data sets to illustrate the wide applicability of the algorithm.

1.2 Related Work

Following the recent successes of the targeted advertising services, there have been several empirical studies that investigate the effects of displaying sponsored content alongside organic content [1, 6, 7]. There have also been attempts to explain such effects through theoretical models [8, 9]. In addition, several researchers have worked on designing systems and algorithms from the content provider's perspective for revenue maximization through efficient auction of the ad-space [10] and from the advertiser's perspective for effectively reaching the target audience [11, 12]. It is empirically shown in [1] that customers are less likely to select recommendations which are tagged as "advertisement" or "sponsored", motivating the advertisers to remove such tags.

Prior work on anomaly detection in recommender systems exists from the perspective of a recommendation engine as a victim of false user-profile injections [13, 14]. To the best of our knowledge, ours is the first work that considers the problem from the users' perspective and proposes a mechanism for detection of bias in recommendation engines.

2 System Model

In this section, we describe our assumptions about the structural properties of *objective* and *biased* recommender systems by the means of a probabilistic model. This model does not include any particulars about the working of the recommendation engine and therefore typifies a broad class of recommender systems. Before we proceed to describe the model in detail, the salient features of this model are listed below:

- An *objective* recommendation engine has a fairly good estimate of the user preferences.

- An *objective* recommendation engine follows the monotonic property – higher preference to higher ranked items.
- A *biased* recommendation engine systematically gives preference to a small set of items irrespective of users' tastes.

Notation: Our notation $O, \Omega, \Theta, o, \omega$ to describe the asymptotics of various parameters with increasing size of the database (total number of items in the database) is according to the standard Landau notation. We say that an event occurs with high probability if the probability of the event tends to 1 as the size of the database goes to infinity. We use $\mathbb{1}\{\cdot\}$ to represent the indicator function, i.e.,

$$\mathbb{1}\{\mathsf{E}\} := \begin{cases} 1 & \text{if event } \mathsf{E} \text{ occurs,} \\ 0 & \text{otherwise.} \end{cases}$$

Equality and inequality between random variables always refer to almost sure (with probability 1) conditions unless otherwise specified. For example, if X and Y are two random variables, then $X = Y$ implies $X = Y$ *a.s.* For any given matrix, $\mathbf{R}$, the u^{th} row of $\mathbf{R}$ is represented by $\mathbf{R}_u$.

2.1 User-Item Database

The recommendation engine recommends products to users from a large database of m items indexed from 1 to m. A user's opinion about an item is represented by a numerical value that we call the user's *rating* of that item. It should be noted that these ratings are only an implicit representation of true opinions of the users – higher the rating, better suited is the item for the user. We denote the user-item rating matrix for the entire database by $\mathbf{R}$, where rows indicate users and columns indicate items. We introduce a parameter called the *efficacy threshold*, denoted by η which is used to represent opinions on a binary scale. We assume that a user is satisfied with a recommendation if the rating of the recommended item is greater than or equal to η. We refer to such a recommendation and item as being *effective* for that user.

Definition 1 (Effective & Ineffective)**.** *An item i is effective for a user u if the rating of that item by the user, R_{ui} is at least η. Similarly, a recommendation is said to be effective for a user if the recommended item is effective. An item or recommendation that is not effective is said to be ineffective.*

Let $f_u(\eta, [m])$ denote the number of items in the database $[m]$ whose rating is greater than or equal to η for user u. In other words, it is the number of effective items in the database for user u.

Let us define the function $F : \mathbb{R} \times \mathbb{R}^m \to \mathbb{R}$ as follows:

$$F(r, \mathbf{R}_u) := |\{i : R_{ui} \geq r\}|,$$

where R_{ui} is the i^{th} element of the m-length vector $\mathbf{R}_u$. This function is used to find the number of items whose rating exceeds value r for any player u if the ratings of all the items

in the database for player u is given by $\mathbf{R}_u$. For example, if $\mathbf{R}_u$ is the row corresponding to player u in the rating matrix, $\mathbf{R}$, then $F(\eta, \mathbf{R}_u)$ is equal to $f_u(\eta, [m])$, the number of effective items for user u. Similarly, $F(R_{ui}, \mathbf{R}_u)$ gives the rank of item i for user u. Also note that for any given $\mathbf{R}_u$, $F(r, \mathbf{R}_u)$ is a non-increasing function of r.

2.2 Recommendation Engine

We next describe the behavior of a recommendation engine using a probabilistic model. Let $\mathbb{1}_{ui}(t)$ indicate whether item i has been recommended to user u at time t, i.e.,

$$\mathbb{1}_{ui}(t) := \begin{cases} 1 & \text{if item } i \text{ is recommended to user } u \text{ at time } t, \\ 0 & \text{otherwise.} \end{cases}$$

We make the following assumption about any recommender system: An item that has been recommended to a user once is not recommended to the same user again, i.e., for any user, u and item, i, $\sum_{t=1}^{\infty} \mathbb{1}_{ui}(t) \leq 1$.

2.2.1 Objective Recommendation Engine

An *objective* recommendation engine is considered to consist of two components - one is the *learning* strategy which estimates the user-item rating matrix by the means of available feedback from users, and another is the *recommendation* strategy which generates recommendations based on the estimated user preferences. Our model does not specify the details of the learning strategy except requiring that the output of the strategy, that is the estimate of the user-item matrix, be close to the original rating matrix, $\mathbf{R}$. Therefore, this model could be applied to a wide class of recommendation engines which estimate users' preferences fairly well. Let the estimate of the rating matrix at time t be denoted by $\hat{\mathbf{R}}(t) = \left[\hat{R}_{ui}(t)\right]$. This estimate is modeled as the sum of the original rating matrix and an additive noise matrix whose elements are independent across users, items and time. This can be written as $\hat{\mathbf{R}}(t) = \mathbf{R} + \boldsymbol{\epsilon}(t)$, where $\boldsymbol{\epsilon}(t) = [\epsilon_{ui}(t)]$ is the noise matrix and $\epsilon_{ui}(t)$ is independent of $\epsilon_{u'i'}(t')$ for all u, u', i, i', t, t'.

The recommendation strategy uses the estimated user-item rating matrix $\hat{\mathbf{R}}(t)$ to make recommendations at time t. The following model characterizes the behavior of an *objective* recommendation strategy:

1. Recommendations are made based on a user-item weight matrix, denoted by $\mathbf{W}(t) = [W_{ui}(t)]$. This is a stochastic matrix (rows sum to one), which is updated based on the current estimate of the rating matrix, $\hat{\mathbf{R}}(t)$.

2. Given the weight matrix, a user is given a recommendation by choosing an item randomly, independent of everything else, with weights given by the row corresponding to the user in the user-item weight matrix.

3. At any time t, the weight matrix $\mathbf{W}(t)$ satisfies the following *monotonic* property: if i and j are two items that have not been shown to user u and the ratings are such that $\hat{R}_{ui}(t) \geq \hat{R}_{uj}(t)$, then the weights satisfy $W_{ui}(t) \geq W_{uj}(t)$.

2.2.2 Biased Recommendation Engine

A *biased* recommendation engine marks a small set of items, $\mathcal{A}$ ($\subseteq [m]$) from the item database as *ads*. To make a recommendation to a user, with probability γ, independent of everything else, it chooses an item that has not been shown from the ad-pool, $\mathcal{A}$. And with probability $1 - \gamma$, it can follow any recommendation algorithm (for example, an *objective* recommendation algorithm). We refer to γ as the *bias probability*. Note that the strategy for showing ad items is unspecified except that no item is shown to a user twice. In particular, the engine may even customize its ad recommendations according to users' tastes. As in the case of the complete database, let $f_u(\eta, \mathcal{A})$ denote the number of effective ads in the ad-pool, $\mathcal{A}$ for user, u.

2.3 Discussion of Assumptions

Some of the assumptions in the recommender system model above are present only for ease of analysis. We discuss below how they can be relaxed in practical settings.

1. It is assumed that, in any recommendation engine, an item once recommended to a user is not recommended to the same user again. This condition is required only to ensure that there are no repeated recommendations of sponsored advertisements that might be effective. Indeed, if all sponsored ad recommendations are effective, it would not be possible to distinguish them from genuine recommendations. This assumption can therefore be relaxed to require sufficient number of ineffective ad recommendations in a *biased* recommender system.

2. The noise in estimation of the user-item rating matrix is assumed to be additive i.i.d. noise. This can be replaced by a more general noise model in which the elements of the estimated user-item matrix are independent across users, items and time. The independence assumption is used to model arbitrary errors which are unlikely to skew the estimated matrix in such a way as to give high preference to a small number of ineffective items uniformly across a large subset of users.

3. We assume that a *biased* recommendation engine decides to show sponsored ads with probability γ (bias probability) independent of everything else. This assumption, again, is used only for ease of exposition. It is sufficient to have arbitrary γ fraction of the recommendations from the ad-pool, not necessarily chosen at random.

3 Anomaly Detection Algorithm and Theoretical Results

In this section, we describe the algorithm for detecting anomalous systems and provide analysis of Type I and Type II errors as in binary hypothesis testing.

3.1 Anomaly Detection System

The problem is to design a test to detect if a recommendation engine is *biased*. In other words, the test has to decide between the following two hypotheses:

- H_1 : "The recommendation engine is *biased*," and
- H_0 : "The recommendation engine is not *biased*."

It is similar to a hypothesis testing problem except that the statistical distribution for the two hypotheses are not well defined. The only *a priori* knowledge that is assumed is the structure of a *biased* recommendation engine as specified in Section 2.2. But the specifics of various parameters in the recommendation engine, such as bias probability γ and the ad-pool $\mathcal{A}$ is unknown. As in traditional hypothesis testing problems, we make use of multiple data points obtained from many users who constitute the *anomaly detection system*. The anomaly detection system consists of a set of n *players* which is a subset of the user database in the recommendation system. These players can give accurate binary feedback (effective or ineffective) on the items recommended to them. Without loss of generality, we denote these players as users indexed from 1 to n in the user database.

3.2 Algorithm

We now describe an algorithm called *Bi*nary feedback *A*nomaly *D*etector (*BiAD*), that uses the recommendations made to the players and their feedback to decide between one of the two hypotheses. In every *round* of recommendation, each player is recommended an item by the recommendation engine. In round t, the algorithm uses the feedback from the players and computes for each item, the total number of players until that round who have been recommended that item and found the item ineffective. This number is denoted by $B_i(t)$ for item i. If the sum of the largest $\hat{A}(t)$ of these numbers among all the items is greater than or equal to a threshold $T(t)$, the recommendation engine is declared to be *biased*. Otherwise, the same procedure is repeated in the next round. If the algorithm does not declare the engine to be *biased* in $Q(m)$ rounds, then the hypothesis that the engine is *biased* is rejected. Here, $\hat{A}(t)$, $T(t)$ and $Q(m)$ in the algorithm are appropriately designed parameters (given by Equations 1-7). The pseudocode for this algorithm is shown in Algorithm 1.

As opposed to the basic average test, this algorithm searches for concentration of large number of ineffective items in a small set. Since the number of potential advertisements is unknown, this algorithm makes decisions in real-time as it gets feedback from the players. Larger the size of the ad-pool, larger is the number of feedback samples required to detect a *biased* engine. (The trade off between various parameters is discussed in detail in Section 4.) Therefore, the algorithm increases the size of the search set with progressing rounds of recommendation. Also, note that the algorithm requires only binary feedback from the players – whether the recommendations are effective or ineffective, which explains the name of the algorithm.

The following theorem gives sufficient conditions for good performance of the algorithm. Unlike in general hypothesis testing problems, we define Type I error, which corresponds to false positives, only for *objective* systems. On the other hand, Type II error is used to refer

Algorithm 1 *Bi*nary feedback *A*nomaly *D*etector (*BiAD*)

Initialize $t = 1$ (round 1).
while $t \leq Q(m)$ **do**
  Compute $B_i(t)$ = number of players who have rated item i ineffective upto round t for all $i \in [m]$.
  Compute $S(t)$ = sum of the largest $\hat{A}(t)$ among $\{B_i(t)\}_{i=1}^{m}$.
  if $S(t) \geq T(t)$ **then**
    Stop and accept H_1.
  else
    $t \leftarrow t + 1$.
  end if
end while
Stop and reject H_1.

to missed detection in the case of a *biased* system. We do not give any guarantees for the class of recommendation engines that are neither *objective* nor *biased*.

Theorem 1. *Let the parameters in the detection algorithm, BiAD satisfy the following equations:*

$$\hat{A}(t) = t, \tag{1}$$

$$T(t) = \exp\left(1 + W\left(\frac{\hat{\beta}(t)}{e}\right)\right)\hat{p}(t), \tag{2}$$

where $W(\cdot)$[1] *represents the Lambert-W or product log function, and*

$$\hat{\beta}(t) = \frac{\left(\hat{A}(t) + c\right)\log m}{\hat{p}(t)} - 1, \tag{3}$$

$$\hat{p}(t) = \exp\left(1 + W\left(\frac{\beta(t)}{e}\right)\right)p(t), \tag{4}$$

$$\beta(t) = \frac{\left(\hat{A}(t) + c\right)\log m}{p(t)} - 1, \tag{5}$$

$$p(t) = \max_{\left\{\hat{\mathcal{A}} \subseteq [m] : \left|\hat{\mathcal{A}}\right| = \hat{A}(t)\right\}} \left\{\sum_{u=1}^{n}\sum_{l=1}^{t} \mathbb{E}\left[P_u^{\hat{\mathcal{A}}}(l)\right]\right\}, \tag{6}$$

$$P_u^{\hat{\mathcal{A}}}(l) = \sum_{i \in \hat{\mathcal{A}}} \frac{\mathbb{1}\left\{R_{ui} < \eta\right\}}{F\left(\hat{R}_{ui}(l), \hat{\mathbf{R}}_u(l)\right) - l + 1}, \tag{7}$$

with $c = 1/2$. *Then BiAD gives the following guarantees on the error probabilities:*

(I) Type I Error:
If the recommendation engine is objective, the probability that BiAD declares it to be anomalous is $O(\frac{Q(m)}{\sqrt{m}})$.

[1]For any $z \in \mathbb{R}$, $W(z)e^{W(z)} = z$.

(II) Type II Error:
If the recommendation engine is anomalous with an ad-pool of size A, and if

(a) the number of ads, $A \leq Q(m)$,

(b) the fraction of recommendations that are ads, i.e., the bias probability $\gamma = \omega\left(\frac{\log m}{n}\right), \omega(\frac{p(A)}{nA})$, and

(c) $\sum_{u=1}^{n} f_u(\eta, \mathcal{A}) = o(\gamma n A)$, where $f_u(\eta, \mathcal{A})$ is the number of effective ads for user u,

then the probability that BiAD does not declare the system as anomalous within A rounds is $e^{-\Omega(\gamma n)}$.

The proof of this theorem is presented in Section 6.

4 Discussion

In this section, we discuss how the error probabilities depend on the parameters of the problem.

4.1 Choice of Threshold

Note that computation of the threshold function, $T(t)$ as specified in Theorem 1 (given by Equation (2)) requires knowledge of the noise statistics and also the players' opinions about all the items in the database. More precisely, since $\hat{\mathbf{R}}(t) = \mathbf{R} + \boldsymbol{\epsilon}(t)$, computation of $\mathbb{E}\left[P_u^{\hat{\mathcal{A}}}(l)\right]$ (see Equation (7)) requires knowledge of $\mathbf{R}_u$ and also the distribution of estimation noise, $\boldsymbol{\epsilon}_u(l)$. The noise statistics reflect the accuracy of the learning strategy of the recommendation engine, and it is possible that these statistics are unknown or cannot be estimated. Moreover, it might also be difficult to obtain the players' opinions about all the items in the database. To overcome this difficulty, a practical implementation of the algorithm could use an approximation of the unknown quantity. We now propose one way to compute such an approximation. Note that

$$\begin{aligned}\mathbb{E}\left[P_u^{\hat{\mathcal{A}}}(l)\right] &= \mathbb{E}\left[\sum_{i\in\hat{\mathcal{A}}} \frac{\mathbb{1}\left\{R_{ui} < \eta\right\}}{F\left(\hat{R}_{ui}(l), \hat{\mathbf{R}}_u(l)\right) - l + 1}\right] \\ &\leq \mathbb{E}\left[\sum_{i\in\hat{\mathcal{A}}} \frac{1}{F\Big(\eta + \epsilon_{ui}(l), \mathbf{R}_u(l) + \boldsymbol{\epsilon}_u(l)\Big) - l + 1}\right],\end{aligned}$$

where the inequality follows since $F\Big(r, \mathbf{R}_u(l) + \boldsymbol{\epsilon}_u(l)\Big)$ is a non-increasing function of r. We assume that the estimates of the ratings are not skewed in one direction, and therefore the noise has zero mean. Since the noise statistics are unknown, we could approximate the right

hand side of the above inequality by substituting the noise term with its mean. With this approximation, the right hand side of the inequality can be substituted with

$$\sum_{i \in \hat{A}} \frac{1}{F\Big(\eta, \mathbf{R}_u(l)\Big) - l + 1} = \frac{\hat{A}(t)}{f_u(\eta, [m]) - l + 1}, \tag{8}$$

where $f_u(\eta, [m])$ is the total number of effective items in the database for user u. Depending on the application, it might be relatively easy to estimate this number or at least estimate a lower bound for this number. As an example, one could roughly estimate that for every user there are $\sqrt{m}$ effective items among the m items in the database. We observe in our simulations that a rough estimate of $f_u(\eta, [m])$ is sufficient to obtain good results.

Note that over-estimation (under-estimation) of $T(t)$ decreases the probability of Type I (Type II) error and increases the probability of Type II (Type I) error. In other words, the higher the value of $T(t)$, the lower is the probability of Type I error and the higher is the probability of Type II error. Therefore, the risk associated with false positives and missed detection could serve as a guideline for the choice of the threshold function. In our simulations (Section 5), we propose a practical threshold function that gives a good balance between the two error probabilities for most scenarios.

4.2 Effect of Parameters on Performance

Theorem 1 gives guarantees on the asymptotic performance of *BiAD* as the size of the database grows large. These guarantees depend on various parameters in the algorithm as well as the recommendation engine. From the analytic bounds derived in Theorem 1, we analyze in this section the trade off between these parameters to understand the conditions under which the algorithm shows good performance. We see that the theoretical results support our intuitive understanding about the conditions under which a *biased* system can be distinguished from an *objective* system. These results are also corroborated by our simulation results described in Section 5.

In Section 4.1, we consider the effect of the choice of the threshold parameter on the error probabilities. We now discuss the effect of other parameters.

Number of Rounds in the Test and Size of the Ad-Pool. It is seen (from Result (I)) that the upper bound on the probability of Type I error increases with increasing number of rounds. This is expected, since it gives more chances to falsely declare a system *biased*. For Type I error to go to zero as the size of the database goes to infinity, it is sufficient if $Q(m) = o(\sqrt{m})$.

Guarantees for detection of a *biased* engine (Result (II)) are dependent on various parameters. One of the conditions is that the ad-pool is not very large (Condition (II)a). Specifically, it is sufficient if the size of the ad-pool, A is at most the maximum number of rounds of recommendations, $Q(m)$. Therefore, increasing $Q(m)$ (the number of rounds of testing) enables detection of larger ad-pools but also increases the probability of Type I error. Intuitively, a small ad-pool conforms with our definition of a *biased* recommendation engine as one that favors a few items over many others and therefore facilitates easier detection.

Number of Effective Ads. For correct detection of a *biased* engine, it is also required that the average number of effective ads (averaged over all players) is not very large (Condition (II)c). A large number of effective ads enables the recommendation system to customize ads according to users' tastes and is contradictory to our interpretation of a *biased* system which recommends ads that do not match with users' preferences.

Number of Players. The dependence on the number of players n is seen in two respects – it determines the minimum bias probability at which detection is guaranteed (Condition (II)b) and also the probability of Type II error. Both these results show that a large number of players improves the prospect of correct identification which can explained by the fact that a large sample size supports better statistical analysis.

Number of Effective Items. The minimum bias probability at which detection is ensured is also determined by the average number of effective items in the entire database. This can be seen from the term $\frac{p(A)}{nA}$ in (Condition (II)b). The no estimation noise case ($\epsilon_{ui}(t) = 0$ for all u, i, t) is useful in understanding the term $\frac{p(A)}{nA}$. When there is no noise, $\frac{p(A)}{nA} = O\left(\frac{1}{nA}\sum_{u=1}^{n}\sum_{l=1}^{A}\frac{A}{f_u(\eta,[m])-l+1}\right)$. Therefore, $\frac{p(A)}{nA}$ has an inverse relation with the number of effective items in the database. This conveys that a large number of effective items facilitates better detection of a *biased* engine. Intuitively, a large number of effective items in the database helps in clearer demarcation of an *objective* engine from a *biased* one. With many effective items in the database, an *objective* system would have a higher probability of recommending effective items, while a *biased* system always makes at least γ fraction of its recommendations from the ad pool where the number of effective ads is limited.

Bias Probability. The more a *biased* engine recommends from the ad-pool, the more apparent is its biased behavior. The fraction of total recommendations that are from the ad-pool is captured by the bias probability, γ. We see that the probability of Type II error decays exponentially with increasing γ, and also that larger γ facilitates easier anomaly detection (Conditions (II)b, (II)c).

Choice of $c = 1/2$. In the course of the proof of Theorem 1, we prove that for any choice of c, the Type I Error is bounded by $O(\,Q(m)\,m^{-c}\,)$. Hence, by changing c in the algorithm, one can control the error probability. However, the downside of increasing c is that it effectively increases the threshold $T(t)$, which results in requiring the bias probability $\gamma = \omega(\log m(1 + (c/A))/n)$.

4.3 Applications

The proposed anomaly detection algorithm is readily applicable in the retail market. It can identify recommender systems that dole out sponsored advertisements in the garb of personalized recommendations. In this era of personalization, there are numerous other applications, two of which are described below. These two examples also illustrate the advantage of *BiAD* in requiring simple binary feedback, allowing it to be applied in a wide variety of scenarios.

Search Engine Bias. Search engine bias is one of the most important ethical issues surrounding search engines, and its social implications have been studied for more than a decade [4, 15, 5]. The Stanford Encyclopedia of Philosophy [16] describes search engine bias as non-neutrality of search engines, where "search algorithms do not use objective criteria" or "favor some values/sites over others in generating their list of results for search queries." A sponsored search engine in the late 1990s called GoTo ranked its search results purely based on bids from advertisers [17]. It was evidently unsuccessful due to users' mistrust of paid searches and was eventually acquired by Google. Google also uses an auction to sell ads but displays them physically separated from organic search results.

The pros and cons of enforcing transparency in the algorithms used for generating search results have been examined in [17, 18]. Even in the absence of total transparency, anomaly detection systems such as *BiAD* could be useful in identifying bias in search engines. With personalization being extended to search results [19, 20, 21], search engines virtually act as recommendation engines. With large number of potential search results, our model of recommender systems with a large database fits well in this problem where biased search engines correspond to *biased* recommender systems. In addition to search engines, this example can be extended to identify hidden sponsored advertisements in social networking sites and online news portals, all of which use personalization algorithms.

Pharmaceutical Lobby. Pharmaceutical lobbying is another controversial issue that affects many parts of the world [22, 23, 24, 25]. Among its many aspects, we focus on the marketing practices of large pharmaceutical companies which manipulate the opinions of doctors, health care providers and law-makers by providing biased information and through other tactics [26, 27]. There have been allegations that big drug companies influence physicians to prescribe their highly priced branded drugs even when other better or cheaper alternatives are available [28, 29].

Again, our interpretation of a *biased* recommendation engine is well-suited to model this scenario. Since drugs are prescribed on a person-to-person basis, health care providers can be viewed as recommendation service providers who recommend drugs to patients, and the lobbying drug companies act as advertisers. A health care system that favors a few incompetent (expensive or ineffective) drugs in spite of the availability of other *better* (cheaper or more effective) alternatives matches well with our definition of a *biased* recommendation engine. With data samples consisting of prescriptions and their efficacy on patients, anomaly detection algorithms like *BiAD* could help watchdog agencies in identifying such malpractices.

5 Numerical Results

We evaluate our algorithm through offline simulations, with careful considerations for ensuring proximity to real world scenarios.

5.1 Simulation Setup

Given below is a detailed description of the methods we adopt in our simulations to replicate the different components of a recommender system.

User-Item Database. Estimating users' opinions about all items in a database is essential for real-world recommender systems, and hence for simulating those recommender systems as well. However, the ground truth on such data set is not available, since in existing data sets each user typically only rates a small subset of items, and those ratings are also noisy and possibly biased.

For a complete user-item rating matrix, we take available sparse data sets ([30, 31, 32]) and renormalize the ratings on a linear scale from zero to ten. The missing entries in the sparse matrix are then filled in using the matrix completion algorithm from [33]. For the purpose of simulating real-world user opinions, we consider this completed matrix as ground truth. We evaluate our algorithm on three data sets:

- D1 a subset of the Amazon cellphones and accessories data set [30] with 3671 users and 8728 items,
- D2 a subset of the Netflix Prize data set [31] with 2951 users and 9259 movies, and
- D3 a subset of the Movielens 10m ratings data set [32] with 3671 users and 8729 movies.

Recommendation Engine. Due to non-transparency of recommendation strategies, it is not exactly known how recommendation engines behave. As a representation of the learning strategies used by these systems, we use two learning algorithms popular in literature:

- L1 Matrix factorization. Specifically, we use the inexact ALM method proposed in [34].
- L2 User-based collaborative filtering (with Pearson correlation as the similarity metric [35]).

To simulate the temporal dynamics of a recommender system, the recommendation engine is initially supplied a sparse subset of the user-item ratings chosen according to a power-law degree distribution observed in real-world data sets [36]. (Specifically, the number of feedback entries from each user is chosen from a $pareto(3, 3)$ distribution.) In each round of recommendation, the engine recommends one item to each user and observes users' feedback about the recommended items. It periodically updates its estimate of the users' preferences based on this feedback. In our experiments, we set the frequency of these updates to once in every 5 rounds.

It is natural for a recommendation engine to have an *explore* component to address the cold start problem and have wider coverage of the database [37]. Therefore, in all our experiments we invoke random explore for 0.1 fraction of the recommendations made. In a recommendation meant for exploration, an item is chosen uniformly at random from the database, provided it has not been shown previously.

For all other recommendations which do not explore, we use the following recommendation strategy – for each user, the items are ranked according to the estimated preferences. To make a recommendation, an *objective* recommendation engine chooses the highest ranked item among the items not yet recommended. The recommendation strategy of a *biased* engine follows the description in Section 2.2.2 – for any recommendation, with probability $1 - \gamma$, like an *objective* engine, it recommends the highest ranked item and with probability γ, it recommends an item from the ad-pool. We consider two kinds of ad selection strategies – from the among the ads that have not already been recommended to the user,

- A1 An ad item is chosen uniformly at random.
- A2 The ad item which has the highest ranking is chosen.

Strategy A2 corresponds to customization of ads according to users' tastes. Note that this

is harder to detect than strategy A1 since it has higher likelihood of recommending effective ads.

Anomaly Detection System. For players in the anomaly detection system, we randomly choose a subset of users from the data set. In experiments which test the performance of the algorithm with increasing number of players, we choose the subset of players incrementally.

As explained in Sections 2 and 3, the algorithm requires feedback samples of efficacy of the recommendations made to the players. For our experiments, we adopt the characterization of efficacy used in our theoretical model given by Definition 1. Note that the number of effective items in the database for any user depends on the efficacy threshold η. To be able to test the algorithm for different number of effective items, we choose a different efficacy threshold for each of the data sets. Specifically, we set $\eta = 5.5, 8.0$, and 8.8 for data sets D1, D2, and D3, which correspond to an average number of effective items of 80, 250, and 150 respectively.

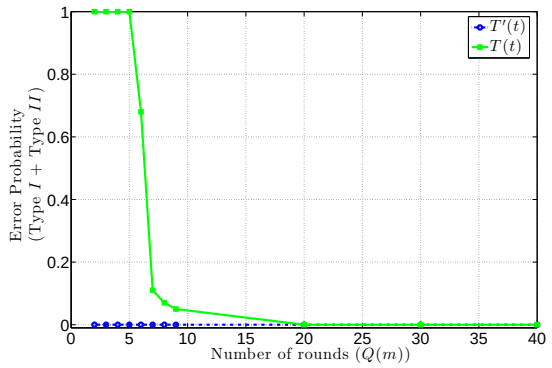
5.2 Results

We evaluate the performance of *BiAD* with variations in different parameters of the recommender system and the anomaly detection system. To demonstrate its effectiveness in different settings, we present performance results for various combinations of data sets (D1-D3) and recommendation algorithms (L1-L2, A1-A2). An *objective* recommendation engine is represented by its learning algorithm (L1 or L2) while a *biased* recommendation engine is represented by its learning algorithm (L1 or L2) and its ad-recommendation strategy (A1 or A2). Although we present experimental results for specific combinations for space limitations, other settings give similar trade offs. Specifically, the simulation results corroborate our theoretical analysis of the tradeoffs between various parameters in Section 4.

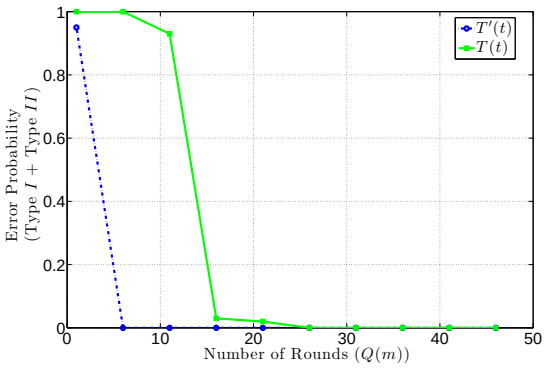
We now describe how the performance depends on the choice of various parameters in *BiAD*. We set the parameter $\hat{A}(t)$ according to Equation (1) in all the simulations. Other parameters of the algorithm are discussed below.

Threshold. As explained in Section 4.1, varying the threshold parameter $T(t)$ in the algorithm affects Type I and Type II error probabilities in opposite ways. Using lower values of threshold increases probability of Type I error and decreases that of Type II error. The threshold given by Equation (2) is designed to ensure, irrespective of the number of players in the anomaly detection system, low probability of false positive (Type I error) even when the estimated user preferences are noisy. This is especially important if the risk associated with false implication of an *objective* recommendation engine is high.

We observe that a less conservative threshold gives a better balance between the two types of errors. Specifically, we use a threshold that can be proved to guarantee low error rates under the assumption that the recommendation engine's estimation of user preferences are accurate. This threshold, denoted by $T'(t)$, is equal to the value of $\hat{p}(t)$ given by Equation (4). In all our simulations, we show the performance of *BiAD* for both these threshold choices. Simulation results show that $T'(t)$ gives better performance than $T(t)$ except in one case (Figure. 2b) where those two choices give similar performances.



(a) Data set : D1 , Algorithm : L2 + A1 , $n = 100, \gamma = 0.45, A = 8$.



(b) Data set : D2 , Algorithm : L2 + A2 , $n = 100, \gamma = 0.35, A = 8$.

Figure 1: Number of rounds in the test, $Q(m)$ affects the number of ads that can be detected at least 8 and 15 rounds required for $T'(t)$ and $T(t)$ respectively.

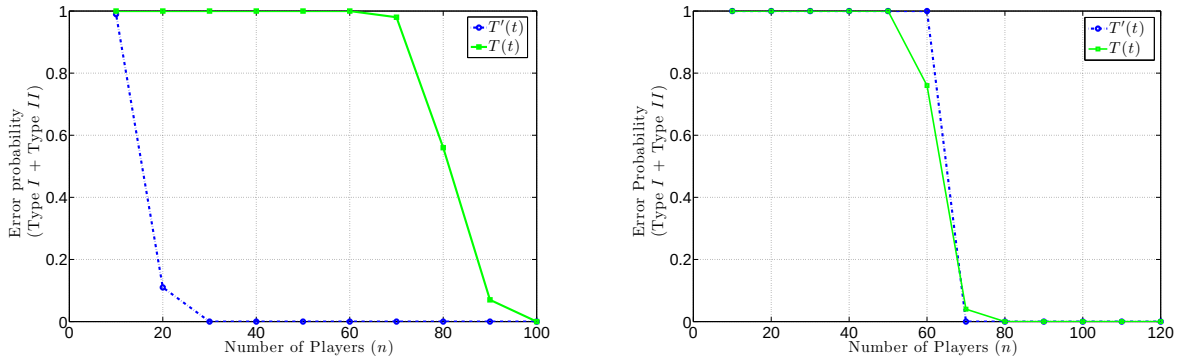
In both these thresholds, $\mathbb{E}[P_u^{\hat{A}}(l)]$ in Equation (6) is substituted with the right hand side of (8). This requires knowledge of the number of effective items for each player in the anomaly detection system. In our simulations, *BiAD* approximates this with the average number of effective items for all the users. Effectively, it uses the following value of $p(t)$ instead of Equation (6):

$$p(t) = \sum_{l=1}^{t} \frac{n\hat{A}(t)}{\tilde{f}([m]) - l + 1}, \tag{9}$$

where $\tilde{f}([m])$ is an estimate of the average number of effective items in the database $[m]$. We assume that this average number is not very difficult to estimate and in all the results, unless specified, *BiAD* has an accurate estimate of this number.

Number of Rounds in the Test. The number of rounds of recommendation $Q(m)$ affects the error probability. This is seen in Figure 1 which shows the variation of sum of Type I and Type II errors with $Q(m)$. Type I error rate is in fact close to zero for all values of $Q(m)$ for both the thresholds, so the plots effectively show Type II error rates. Theorem 1 guarantees detection of a *biased* engine if $Q(m) \geq A$. The plots show that *BiAD* detects 8 ads if $Q(m)$ is at least 8 and 15 for $T'(t)$ and $T(t)$ respectively. For all the remaining simulations, we set the parameter $Q(m) = 40$.

Number of Players. Larger number of players in the anomaly detection system indicates higher number of input samples to the algorithm, and as expected, the algorithm performs better as this number increases. In Figure 2, we plot the sum of Type I and Type II error rates with increasing number of users. To detect a *biased* engine with the specified value of γ, these plots show that 70 and 100 players respectively are sufficient when $T'(t)$ and $T(t)$ are chosen to be the threshold parameter. We use 100 players in all other simulations.



(a) Data set : D1, Algorithm : L1 + A1, $A = 8, \gamma = 0.45$.

(b) Data set : D2, Algorithm : L1 + A1 , $A = 8, \gamma = 0.35$.

Figure 2: The performance improves with number of collaborating users n.

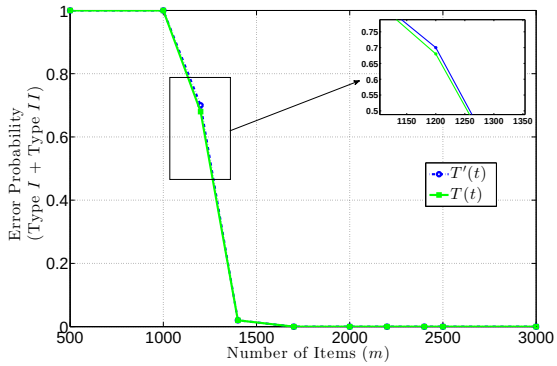
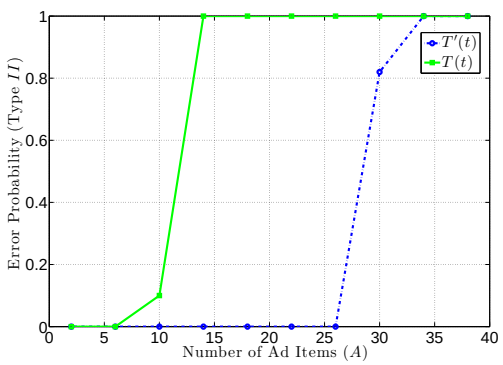


Figure 3: Variation of Type I + Type II error rates with size of data set. Data set : D2, Algorithm : L1 + A2, $A = 8, \gamma = 0.35, n = 100$.. When there are more choices to recommend, the user satisfaction with *objective* recommender systems improves making detection easier.

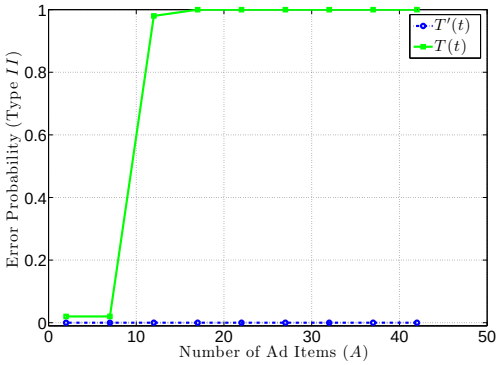
In addition to the choice of parameters in the algorithm, various aspects of the recommender system affect the performance of *BiAD*. These are described below.

Size of the Database. Theorem 1 shows that *BiAD* performs well for recommender systems with large item databases. Databases of varying size are constructed by sub-sampling items from the original data set. Figure 3 shows the variation of Type I and Type II errors with the size of the database. $T(t)$ and $T'(t)$ have very similar performance for the parameters in this experiment. The plot shows that, for detection of 8 ads recommended 35 percent of the time, the algorithm is effective for databases of size 1500 items or larger.

We now demonstrate that *BiAD* has been appropriately designed to identify *biased* engines that systematically recommend (make a sizable fraction of recommendations) from a small ad-pool.



(a) Data set : D1 , Algorithm : L1 + A1 , $n = 100, \gamma = 0.45$.



(b) Data set : D2 , Algorithm : L2 + A2 , $n = 100, \gamma = 0.35$.

Figure 4: As the size of the ad-pool A increases, the (personalized) ads become similar to effective recommendations, making it hard to detect (Type II error is large).

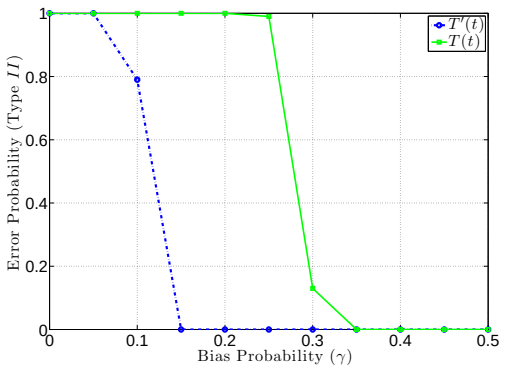
Size of the Ad-Pool. Theorem 1 shows that *BiAD* guarantees detection of a *biased* engine that has a small ad-pool. This same effect is also observed in simulations – Figure 4 shows rate of missed detection (Type II error rate) with varying size of the ad-pool. It is seen that both the thresholds perform well for small number of ads, while threshold $T'(t)$ can detect an ad-pool of size upto 25.

Bias Probability. The bias probability γ quantifies the intensity of bias of the recommendation engine. Plots (Figure 5) for Type II error rate with γ show that *more biased* (higher γ) engines are easier to detect.

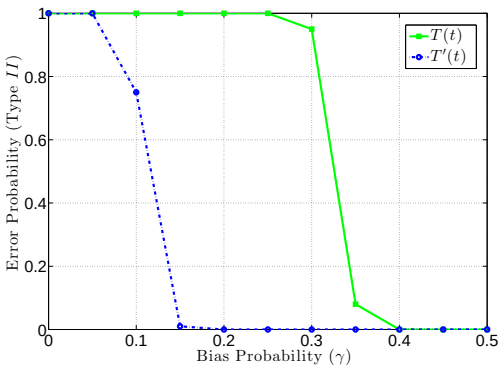
Estimate of the Number of Effective Items. In all the simulations above, it is assumed that *BiAD* has an accurate estimate of the average number of effective items ($\tilde{f}([m])$) which is used to determine the threshold parameter in the algorithm (See Equation (9)). Note that overestimation of this parameter lowers the threshold parameter thereby increasing the probability of Type I error and decreasing the probability of Type II error. Figure 6 shows the effect of variations in this estimate for data set D3 which has an average of 150 effective items. We observe that $T'(t)$ performs well for a wide range of estimates. In the case of $T(t)$, it is safer to overestimate the parameter $\tilde{f}([m])$ than to underestimate it.

5.3 Ineffectiveness of Basic Average Test

As explained in the introduction (Section 1), we demonstrate the inability of the basic average test to distinguish between random errors and deliberate promotion of ads. This test computes the average rating across all recommendations and decides between the two hypotheses based on a threshold parameter. With the specifics of the recommendation strategy (explore probability) unknown, it is difficult to estimate the right value of threshold. For an explore probability of 0.1, Figure 7 shows the performance of the basic average test for different values of the threshold, denoted τ. It is seen that threshold values around 3 give



(a) Data set : D2 , Algorithm : L1 + A2 , $n = 100, A = 8$.



(b) Data set : D3 , Algorithm : L1 + A2 , $n = 100, A = 8$.

Figure 5: Type II error rate decreases as bias probability γ increases.

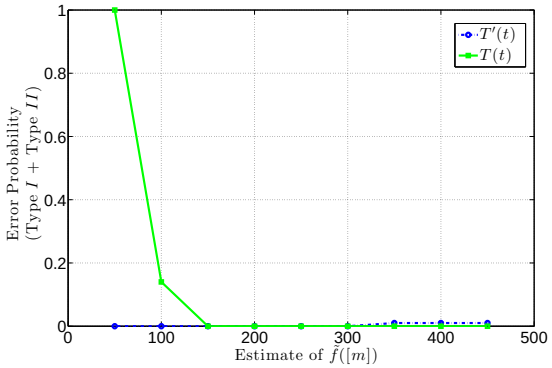


Figure 6: Variation of Type I + Type II error rates with perturbations in the algorithm's estimate of the average number of effective items $\tilde{f}([m])$. Data set : D3, Algorithm : L1 + A1, $A = 8, \gamma = 0.4, n = 100$.

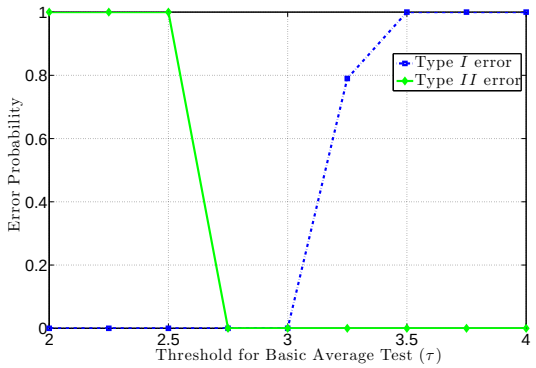


Figure 7: Variation of Type I and Type II error rates with threshold τ for the basic average test shows that the naive approach is sensitive to the choice of parameter τ. Data set : D1, Algorithm : L1 + A1, $A = 8, \gamma = 0.45, n = 100$, Explore Probability = 0.1.

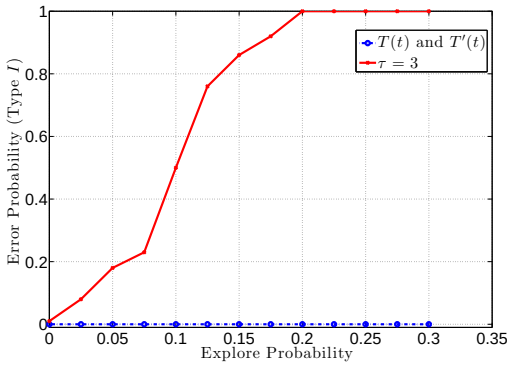


Figure 8: Variation of Type I error rates with variation in explore probability shows that the threshold for the basic average test is sensitive the value of explore probability. Data set : D1, Algorithm : L1, $n = 100$.

the best performance. But, as shown in Figure 8, this same threshold value fails for other values of explore probability. For example, the basic average test falsely declares an *objective* recommendation engine with 20 percent explore probability as *biased*. This shows that the correct choice of τ is sensitive to the explore probability. In contrast, note that *BiAD* has nearly zero Type I error rate for all values of explore probabilities.

6 Proofs

Before we provide the proof of the main theorem, we state and prove two lemmas used in proving the main theorem.

Lemma 2. *For any* objective *recommendation algorithm and for any user u, item i, and time $t < (F(\hat{R}_{ui}(t), \hat{\mathbf{R}}_u(t)) + 1)$, the probability that item i is recommended to user u at time*

t is upper bounded by

$$W_{ui}(t) \leq \frac{1}{F\left(\hat{R}_{ui}(t), \hat{\mathbf{R}}_u(t)\right) - t + 1}\,. \tag{10}$$

Proof.

$$\begin{aligned}
1 &\overset{(a)}{=} \sum_{j=1}^{m} W_{uj}(t) \\
&\geq \sum_{j=1}^{m} \mathbb{1}\left\{\hat{R}_{uj}(t) \geq \hat{R}_{ui}(t)\right\} \left(1 - \sum_{l=1}^{t-1} \mathbb{1}_{ui}(l)\right) W_{uj}(t) \\
&\overset{(b)}{\geq} \sum_{j=1}^{m} \mathbb{1}\left\{\hat{R}_{uj}(t) \geq \hat{R}_{ui}(t)\right\} \left(1 - \sum_{l=1}^{t-1} \mathbb{1}_{ui}(l)\right) W_{ui}(t) \\
&= \left|\left\{j : \hat{R}_{uj}(t) \geq \hat{R}_{ui}(t),\, \sum_{l=1}^{t-1} \mathbb{1}_{uj}(l) = 0\right\}\right| W_{ui}(t) \\
&\geq \left(\left|\left\{j : \hat{R}_{uj}(t) \geq \hat{R}_{ui}(t)\right\}\right| - \left|\left\{j : \sum_{l=1}^{t-1} \mathbb{1}_{uj}(l) \neq 0\right\}\right|\right) W_{ui}(t) \\
&= \left(F\left(\hat{R}_{ui}(t), \hat{\mathbf{R}}_u(t)\right) - (t-1)\right) W_{ui}(t),
\end{aligned}$$

where the (a) and (b) follow from the characterization of an *objective* recommendation algorithm in Section 2.2.1 – equality (a) due to the fact that $\mathbf{W}(t)$ is a stochastic matrix (Property 1), and inequality (b) due to the monotonic property satisfied by the weight matrix (Property 3). The above inequality gives the desired bound in (10) for $t < (F(\hat{R}_{ui}(t), \hat{\mathbf{R}}_u(t)) + 1)$.

Lemma 3. *Let* $\{X_i,\, i = 1, \ldots, k\}$ *be independent Bernoulli random variables with mean* $\{p_i,\, i = 1, \ldots, k\}$*, and let* $\sum_{i=1}^{k} p_i \leq p$*. Then,*

$$\mathbb{P}\left[\sum_{i=1}^{k} X_i \geq T\right] \leq \exp\left(-T \log\left(\frac{T}{p}\right) + T - p\right) \; \forall\, T > p.$$

Using Chernoff bound for independent random variables, we have, for any $\theta > 0$,

$$\begin{aligned}
\mathbb{P}\left[\sum_{i=1}^{k} X_i \geq T\right] &\leq e^{-\theta T} \prod_{i=1}^{k} \mathbb{E}\left[e^{\theta X_i}\right] \\
&= e^{-\theta T} \prod_{i=1}^{k} \left(p_i e^{\theta} + 1 - p_i\right) \\
&\leq e^{-\theta T} \left(\frac{1}{k} \sum_{i=1}^{k} \left(p_i e^{\theta} + 1 - p_i\right)\right)^k \\
&\leq e^{-\theta T} \left(\frac{p}{k}\left(e^{\theta} - 1\right) + 1\right)^k,
\end{aligned}$$

where the second inequality follows from the fact that the geometric mean of non-negative numbers is at most their arithmetic mean, and the last inequality follows for $\theta > 0$, which is true for the choice of $\theta = \log\left((k-p)T/(p(k-T))\right)$ for any $T > p$. Then, we get

$$\begin{aligned}
\mathbb{P}\left[\sum_{i=1}^{k} X_i \geq T\right] &\leq \left(\frac{p(k-T)}{(k-p)T}\right)^T \left(\frac{(k-p)T}{(k-T)k} + 1 - \frac{p}{k}\right)^k \\
&= \left(\frac{p}{T}\right)^T \left(\frac{k-p}{k-T}\right)^{k-T} \\
&= \left(\frac{p}{T}\right)^T \left(1 + \frac{T-p}{k-T}\right)^{k-T} \\
&\leq \left(\frac{p}{T}\right)^T e^{T-p} \\
&= \exp\left(-T\log\left(\frac{T}{p}\right) + T - p\right).
\end{aligned}$$

□

Proof of Theorem 1. The proof of the theorem consists of two parts which give upper bounds for probability of Type I and Type II errors.

Type I Error

The algorithm makes a Type I error if it declares an *objective* recommendation engine to be *biased*. This section shows that the probability that the algorithm makes Type I Error is low.

Suppose that the recommendation engine uses an *objective* recommendation algorithm. We first bound the probability that *BiAD* accepts H_1 in round t. Recall that the algorithm accepts H_1 in round t if $S(t) \geq T(t)$, and that

$$S(t) = \max_{\left\{\hat{\mathcal{A}} \subseteq [m] : \left|\hat{\mathcal{A}}\right| = \hat{A}(t)\right\}} \sum_{i \in \hat{\mathcal{A}}} B_i(t).$$

Consider a fixed $\hat{\mathcal{A}} \subseteq [m]$ such that $|\hat{\mathcal{A}}| = \hat{A}(t)$. We first bound the probability that $\sum_{i\in\hat{\mathcal{A}}} B_i(t) \geq T(t)$ and then use union bound over all possible $\hat{\mathcal{A}}$ to obtain an upper bound on the probability that $S(t) \geq T(t)$. In this direction, we define

$$X_u(l) := \sum_{i \in \hat{\mathcal{A}}} \mathbb{1}_{ui}(l) \cdot \mathbb{1}\left\{R_{ui} < \eta\right\}. \tag{11}$$

Note that $X_u(l)$ is equal to 1 if in round l, player u is recommended some item from set $\hat{\mathcal{A}}$ that is not effective and is 0 otherwise. Given $\left\{\mathbf{W}(l), \hat{\mathbf{R}}(l),\, l \in [t]\right\}$, we have that $\{X_u(l),\, l \in [t],\, u \in [n]\}$ are Bernoulli random variables independent of each other. For every $1 \leq u \leq n$ and $1 \leq l \leq t$, the mean of $X_u(l)$ can be bounded as follows:

$$\begin{aligned}
\mathbb{E}\left[X_u(l) \middle| \left\{\mathbf{W}(l'), \hat{\mathbf{R}}(l')\right\}_{l'=1}^{t}\right] &= \sum_{i \in \hat{\mathcal{A}}} W_{ui}(l) \mathbb{1}\left\{R_{ui} < \eta\right\} \\
&\leq P_u^{\hat{\mathcal{A}}}(l),
\end{aligned} \tag{12}$$

where the inequality follows from the upper bound for $W_{ui}(l)$ from Lemma 2 and the definition of $P_u^{\hat{\mathcal{A}}}(l)$ ((7)). From (6), we have

$$\sum_{u=1}^{n}\sum_{l=1}^{t}\mathbb{E}\left[P_u^{\hat{\mathcal{A}}}(l)\right] \leq p(t).$$

Note that, since the noise in the rating estimates are independent across time and users, $\left\{P_u^{\hat{\mathcal{A}}}(l),\, l\in[t],\, u\in[n]\right\}$ are independent random variables. We now use the Chernoff bound in Lemma 3 to obtain a probabilistic upper bound on their sum.

$$\mathbb{P}\left[\sum_{u=1}^{n}\sum_{l=1}^{t}P_u^{\hat{\mathcal{A}}}(l)\geq\hat{p}(t)\right]\leq\exp\left(-\hat{p}(t)\left(\log\left(\frac{\hat{p}(t)}{p(t)}\right)-1\right)-p(t)\right).$$

By the definition of Lambert-W function,

$$\hat{p}(t)\left(\log\left(\frac{\hat{p}(t)}{p(t)}\right)-1\right)=\left(\hat{A}(t)+c\right)\log m-p(t),$$

which further implies that

$$\mathbb{P}\left[\sum_{u=1}^{n}\sum_{l=1}^{t}P_u^{\hat{\mathcal{A}}}(l)\geq\hat{p}(t)\right]\leq\exp\left(-\left(\hat{A}(t)+c\right)\log m\right). \tag{13}$$

We now proceed to obtain a probabilistic upper bound for the sum, $\sum_{u=1}^{n}\sum_{l=1}^{t}X_u(l)$. By inequality (12), the sum of the corresponding means has the following upper bound:

$$\sum_{u=1}^{n}\sum_{l=1}^{t}\mathbb{E}\left[X_u(l)\middle|\left\{\mathbf{W}(l'),\hat{\mathbf{R}}(l')\right\}_{l'=1}^{t}\right]\leq\sum_{u=1}^{n}\sum_{l=1}^{t}P_u^{\hat{\mathcal{A}}}(l).$$

Since $\{X_u(l),\, l\in[t],\, u\in[n]\}$ are independent Bernoulli random variables given $\left\{\mathbf{W}(l),\hat{\mathbf{R}}(l),\, l\in[t]\right\}$, Lemma 3 can be used as before. If $\sum_{u=1}^{n}\sum_{l=1}^{t}P_u^{\hat{\mathcal{A}}}(l)\leq\hat{p}(t)$, then Lemma 3 gives

$$\mathbb{P}\left[\sum_{u=1}^{n}\sum_{l=1}^{t}X_u(l)\geq T(t)\middle|\left\{\mathbf{W}(l'),\hat{\mathbf{R}}(l')\right\}_{l'=1}^{t}\right]\leq\exp\left(-\left(\hat{A}(t)+c\right)\log m\right). \tag{14}$$

The above upper bound can be derived in exactly the same manner as the upper bound in (13). Combining this upper bound with inequality (13), we have

$$\begin{aligned}\mathbb{P}\left[\sum_{u=1}^{n}\sum_{l=1}^{t}X_u(l)\geq T(t)\right]&\leq\mathbb{P}\left[\sum_{u=1}^{n}\sum_{l=1}^{t}X_u(l)\geq T(t)\middle|\sum_{u=1}^{n}\sum_{l=1}^{t}P_u^{\hat{\mathcal{A}}}(l)\leq\hat{p}(t)\right]+\mathbb{P}\left[\sum_{u=1}^{n}\sum_{l=1}^{t}P_u^{\hat{\mathcal{A}}}(l)\geq\hat{p}(t)\right]\\&\leq 2\exp\left(-\left(\hat{A}(t)+c\right)\log m\right).\end{aligned} \tag{15}$$

Now, recall that $B_i(t)$ is the number of players who have rated item i ineffective upto round t, which can be mathematically written as $B_i(t) = \sum_{u=1}^{n} \sum_{l=1}^{t} \mathbb{1}_{ui}(l) \cdot \mathbb{1}\{R_{ui} < \eta\}$. Using definition (11), we have

$$\sum_{i \in \hat{\mathcal{A}}} B_i(t) = \sum_{u=1}^{n} \sum_{l=1}^{t} X_u(l),$$

which gives us the following equivalent form of inequality (15):

$$\mathbb{P}\left[\sum_{i \in \hat{\mathcal{A}}} B_i(t) \geq T(t)\right] \leq 2 \exp\left(-\left(\hat{A}(t) + c\right) \log m\right). \tag{16}$$

We can now take a union bound over all possible $\hat{\mathcal{A}}$ to bound the probability that *BiAD* accepts H_1 in round t.

$$\begin{aligned}
\mathbb{P}\left[S(t) \geq T(t)\right] &= \mathbb{P}\left[\bigcup_{\left\{\hat{\mathcal{A}} \subseteq [m] : \left|\hat{\mathcal{A}}\right| = \hat{A}(t)\right\}} \left\{\sum_{i \in \hat{\mathcal{A}}} B_i(t) \geq T(t)\right\}\right] \\
&\leq \sum_{\left\{\hat{\mathcal{A}} \subseteq [m] : \left|\hat{\mathcal{A}}\right| = \hat{A}(t)\right\}} \mathbb{P}\left[\sum_{i \in \hat{\mathcal{A}}} B_i(t) \geq T(t)\right] \\
&\leq 2 \binom{m}{\hat{A}(t)} \exp\left(-\left(\hat{A}(t) + c\right) \log m\right) \\
&\leq 2 \exp\left(-c \log m\right) \\
&= 2m^{-c},
\end{aligned}$$

where the second inequality follows from (16). Further taking a union bound over all rounds,

$$\begin{aligned}
\mathbb{P}[\text{Type } I \text{ Error}] &= \mathbb{P}\left[\cup_{t=1}^{Q(m)} S(t) \geq T(t)\right] \\
&\leq \sum_{t=1}^{Q(m)} \mathbb{P}\left[S(t) \geq T(t)\right] \\
&\leq 2\, Q(m)\, m^{-c}.
\end{aligned}$$

This shows that *BiAD* declares an *objective* recommendation engine as *biased* with probability $O(\frac{Q(m)}{\sqrt{m}})$ for the choice of $c = 1/2$.

Type II Error

The algorithm makes a Type II error if it does not detect an *biased* recommendation engine, i.e., it declares H_0 when H_1 is true. Suppose that the recommendation engine is *biased* with an ad-pool $\mathcal{A}$ of size $|\mathcal{A}| = A$. Fix a $\delta \in (0, 1)$. We prove that $\sum_{u=1}^{n} \sum_{l=1}^{A} \sum_{i \in \mathcal{A}} \mathbb{1}_{ui}(l)$, which is equal to the number of ad recommendations to the n players until round A is at least $\gamma(1-\delta)nA$ with high probability. For any $1 \leq u \leq n$, let $Y_u(l) = 1$ if the *biased* recommendation

engine decides to recommend from the ad-pool to user u in round t. Note that $\sum_{i\in\mathcal{A}} \mathbb{1}_{ui}(l) \geq Y_u(l)$ and that $\{Y_u(l),\, l \in [A],\, u \in [n]\}$ are i.i.d. Bernoulli random variables with mean γ. This gives us

$$\begin{aligned}\mathbb{P}\left[\sum_{u=1}^{n}\sum_{l=1}^{A}\sum_{i\in\mathcal{A}} \mathbb{1}_{ui}(l) < \gamma(1-\delta)nA\right] &\leq \mathbb{P}\left[\sum_{u=1}^{n}\sum_{l=1}^{A} Y_u(l) < \gamma(1-\delta)nA\right] \\ &\leq e^{\left(-\frac{\delta^2\gamma nA}{2}\right)}, \end{aligned} \tag{17}$$

where the last inequality follows from a version of Chernoff bound given in [38] for sum of i.i.d. Bernoulli random variables.

The detection algorithm makes the correct decision if in round t, $S(t) \geq T(t)$ for some $t \leq Q(m)$. We show that $S(A) \geq T(A)$ with high probability. Since $A \leq Q(m)$, the algorithm makes the correct decision with high probability. Now, suppose that the number of ad recommendations to the n players until round A is at least $\gamma(1-\delta)nA$. Since the total number of effective ads to the n players is $o(\gamma nA)$, the total number of ineffective recommendations from the ad-pool until round A is $\gamma\Omega(nA)$. Consequently,

$$\begin{aligned} S(A) &= \max_{\left\{\hat{\mathcal{A}}\subseteq[m]:\left|\hat{\mathcal{A}}\right|=A\right\}} \sum_{i\in\hat{\mathcal{A}}} B_i(t) \\ &\geq \sum_{i\in\mathcal{A}} B_i(t) \\ &\geq \gamma(1-\delta)nA - o(\gamma nA) \\ &= \gamma\Omega(nA). \end{aligned} \tag{18}$$

To prove that $T(A)$ does not exceed the right hand side of inequality (18), we consider the following cases:

(i): $\hat{\beta}(A) \geq e$

Since $W(\cdot)$ is an increasing function in $[0,\infty)$, we have $W\left(\frac{\hat{\beta}(A)}{e}\right) \geq W(1) > \frac{1}{2}$. Now,

$$\begin{aligned} T(A) &= \exp\left(1 + W\left(\frac{\hat{\beta}(A)}{e}\right)\right)\hat{p}(A) \\ &= \frac{\hat{\beta}(A)}{W\left(\frac{\hat{\beta}(A)}{e}\right)}\hat{p}(A) \\ &\leq \frac{(A+c)\log m}{W(1)} \\ &= \frac{\log m}{n}O(nA), \end{aligned}$$

where the second equality follows from the definition of the Lambert W function, and the last inequality follows by using the definition of $\hat{\beta}(A)$ given by (3).

(ii): $\hat{\beta}(A) < e,\ \beta(A) \geq e$

$$\begin{aligned}
\hat{p}(A) &= \exp\left(1 + W\left(\frac{\beta(A)}{e}\right)\right) p(A) \\
&= \frac{\beta(A)}{W\left(\frac{\beta(A)}{e}\right)} p(A) \\
&\leq \frac{(A+c)\log m}{W(1)}. \\
T(A) &= \exp\left(1 + W\left(\frac{\hat{\beta}(A)}{e}\right)\right) \hat{p}(A) \\
&\leq \exp\left(1 + W(1)\right) \hat{p}(A) \\
&= \frac{\log m}{n} O(nA).
\end{aligned}$$

(iii): $\hat{\beta}(A) < e,\ \beta(A) < e$

$$\begin{aligned}
T(A) &= \exp\left(1 + W\left(\frac{\hat{\beta}(A)}{e}\right)\right) \hat{p}(A) \\
&\leq \exp\left(1 + W(1)\right) \hat{p}(A) \\
&\leq \exp\left(2 + 2W(1)\right) p(A) \\
&= \frac{p(A)}{nA} O(nA).
\end{aligned}$$

Since $\gamma = \omega\left(\frac{\log m}{n}\right), \omega(\frac{p(A)}{nA})$, combining the results from the above three cases with inequality (18) gives that $S(A) \geq T(A)$ for m large enough. Therefore, the algorithm declares the correct hypothesis in round A if the number of ad recommendations to the n players until round A is at least $\gamma(1-\delta)nA$. We can therefore use the concentration inequality in (17) to bound the probability of Type II error.

$$\begin{aligned}
\mathbb{P}[\text{Type } II \text{ Error}] &\leq \mathbb{P}\left[S(A) < T(A)\right] \\
&\leq \mathbb{P}\left[\sum_{u=1}^{n}\sum_{l=1}^{A}\sum_{i\in\mathcal{A}} \mathbb{1}_{ui}(l) < \gamma(1-\delta)nA\right] \\
&\leq \exp\left(-\frac{\delta^2}{2}\gamma nA\right) \\
&= e^{-\Omega(\gamma n)}.
\end{aligned}$$

This shows that the probability of Type II error decays exponentially with the number of players and the bias probability. □

7 Conclusion

We propose an algorithm that can identify an *biased* recommendation engine that systematically favors a few sponsored advertisements over other genuine recommendations. We formulate a probabilistic model for recommender systems and give theoretical guarantees for our detection algorithm based on this model. Specifically, we show that the probability of missed detection and false positives are low for recommender systems with large databases. We show through simulations that the algorithm performs well for many data sets and different types of recommendation algorithms. In an age when both personalization and advertising have become very prevalent, this kind of anomaly detection algorithm is relevant in a wide variety of scenarios. We demonstrate how our detection algorithm can be applied to problems such as identification of search engine bias and pharmaceutical lobbying. It would be interesting to investigate ways of deploying such an anomaly detection mechanism in practical settings.

Acknowledgments

This work was supported by the NSF grant CNS-1320175 and the ARO grant W911NF-14-1-0387.

References

[1] J. Beel, S. Langer, and M. Genzmehr, "Sponsored vs. organic (research paper) recommendations and the impact of labeling," in *Research and Advanced Technology for Digital Libraries*. Springer, 2013, pp. 391–395.

[2] R. Albergotti, "Facebook to clean up news feeds," *The Wall Street Journal*, 2014.

[3] R. M. Entman, "Framing bias: Media in the distribution of power," *Journal of communication*, vol. 57, no. 1, pp. 163–173, 2007.

[4] L. D. Introna and H. Nissenbaum, "Shaping the web: Why the politics of search engines matters," *The information society*, vol. 16, no. 3, pp. 169–185, 2000.

[5] R. Epstein, "How google could end democracy," *U.S. News & World Report*, 2014.

[6] A. Ghose and S. Yang, "An empirical analysis of search engine advertising: Sponsored search in electronic markets," *Management Science*, vol. 55, no. 10, pp. 1605–1622, 2009.

[7] C. Tucker, "Social advertising," *SSRN eLibrary*, 2012.

[8] S. Yang and A. Ghose, "Analyzing the relationship between organic and sponsored search advertising: Positive, negative, or zero interdependence?" *Marketing Science*, vol. 29, no. 4, pp. 602–623, 2010.

[9] D. Bergemann and A. Bonatti, "Targeting in advertising markets: implications for offline versus online media," *The RAND Journal of Economics*, vol. 42, no. 3, pp. 417–443, 2011.

[10] S. Lahaie, D. M. Pennock, A. Saberi, and R. V. Vohra, "Sponsored search auctions," *Algorithmic game theory*, pp. 699–716, 2007.

[11] P. Rusmevichientong and D. P. Williamson, "An adaptive algorithm for selecting profitable keywords for search-based advertising services," in *Proceedings of the 7th ACM Conference on Electronic Commerce*. ACM, 2006, pp. 260–269.

[12] A. Turpin and J. Katz, "System and method for implementing advertising in an online social network," Aug. 7 2008, uS Patent App. 12/011,880.

[13] R. Burke, B. Mobasher, C. Williams, and R. Bhaumik, "Classification features for attack detection in collaborative recommender systems," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, pp. 542–547.

[14] B. Mobasher, R. Burke, R. Bhaumik, and C. Williams, "Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness," *ACM Transactions on Internet Technology (TOIT)*, vol. 7, no. 4, p. 23, 2007.

[15] M. Zimmer, "The value implications of the practice of paid search," *Bulletin of the American Society for Information Science and Technology*, vol. 32, no. 2, pp. 23–25, 2006.

[16] H. Tavani, "Search engines and ethics," in *The Stanford Encyclopedia of Philosophy*, spring 2014 ed., E. N. Zalta, Ed., 2014.

[17] D. Elgesem, "Search engines and the public use of reason," *Ethics and information Technology*, vol. 10, no. 4, pp. 233–242, 2008.

[18] L. A. Granka, "The politics of search: A decade retrospective," *The Information Society*, vol. 26, no. 5, pp. 364–374, 2010.

[19] Z. Dou, R. Song, and J.-R. Wen, "A large-scale evaluation and analysis of personalized search strategies," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 581–590.

[20] B. Horling and M. Kulick, "Personalized search for everyone," Google Official Blog, 2009.

[21] A. Crook and S. Ahari, "Personalized search for everyone," bing blog, 2011.

[22] J. Abraham, "The pharmaceutical industry as a political player," *The Lancet*, vol. 360, no. 9344, pp. 1498–1502, 2002.

[23] S. H. Landers and A. R. Sehgal, "Health care lobbying in the united states," *The American journal of medicine*, vol. 116, no. 7, pp. 474–477, 2004.

[24] T. L. Editorial, "The uk drug industry: responsible, ethical, and professional?" *The Lancet*, vol. 366, no. 9500, p. 1828, 2005.

[25] A. Shah, "Pharmaceutical corporations and medical research," *Global Issues*.

[26] D. Blumenthal, "Doctors and drug companies," *New England Journal of Medicine*, vol. 351, no. 18, pp. 1885–1890, 2004.

[27] A. Fugh-Berman and S. Ahari, "Following the script: how drug reps make friends and influence doctors," *PLoS Medicine*, vol. 4, no. 4, p. e150, 2007.

[28] C. S. Landefeld and M. A. Steinman, "The neurontin legacy–marketing through misinformation and manipulation," *New England Journal of Medicine*, vol. 360, no. 2, p. 103, 2009.

[29] B. Goldacre, *Bad Pharma: How drug companies mislead doctors and harm patients*. Macmillan, 2014.

[30] J. McAuley and J. Leskovec, "Hidden factors and hidden topics: understanding rating dimensions with review text," in *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 2013, pp. 165–172.

[31] J. Bennett and S. Lanning, "The netflix prize," in *Proceedings of KDD cup and workshop*, vol. 2007, 2007, p. 35.

[32] "Movielens dataset."

[33] R. H. Keshavan, A. Montanari, and S. Oh, "Matrix completion from a few entries," *Information Theory, IEEE Transactions on*, vol. 56, no. 6, pp. 2980–2998, 2010.

[34] Z. Lin, M. Chen, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," *arXiv preprint arXiv:1009.5055*, 2010.

[35] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: an open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM conference on Computer supported cooperative work*. ACM, 1994, pp. 175–186.

[36] Z. Huang, D. D. Zeng, and H. Chen, "Analyzing consumer-product graphs: Empirical findings and applications in recommender systems," *Management science*, vol. 53, no. 7, pp. 1146–1164, 2007.

[37] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proceedings of the 19th international conference on World wide web*. ACM, 2010, pp. 661–670.

[38] M. Mitzenmacher and E. Upfal, *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge University Press, 2005.